25 | 系统架构：如何从写代码的程序员，成长为软件系统架构师？

2020-07-13 臧萌

职场求生攻略 进入课程 >

讲述：臧萌
时长 13:12 大小 12.09M

你好，我是臧萌。这一节，我们来聊聊软件系统架构师。软件系统架构师简称架构师，它可以说是每个做技术的程序员心向往之的职位。这个职位对于很多人，还有些许神秘的味道。在这节课里，我就来和你谈谈我眼中的软件系统架构师，它其实并没有那么神秘。

软件系统架构师是干什么的

架构师的输入

软件系统架构师的工作，有两个输入。一个输入是要解决的问题，这里说的问题，可能是一个系统，可能是某一类相似的业务等，我们也可以把它认为是一个产品。另一个输入是为了解决这个问题，所能使用的资源。这里说的资源，包括系统的工期、团队和公司的技术储备。以及人才储备、业界可以使用的技术、公司的基础设施等等。

也就是说，在架构师的输入里，一手是问题，一手是资源。当然，这两个东西不是一步到位的，不会有人准备好送给架构师，而是架构师要自己去深入了解和摸索。

架构师解决问题的理念

随着对问题理解的深入，架构师会在脑子里慢慢形成自己对问题的理解。这时候，架构师作用中最重要的一点就会慢慢形成，那就是架构师解决问题的理念。换句话说，就是架构师如何定义这个要被解决的问题。

问题本身要尽量客观地去认识，但是如何看待和定义一个问题，就不是一个客观的事情了。这会受到每个人的理念影响。其实对一个问题的定义，很大程度上就决定了如何解决一个问题。架构师看问题的理念不同，也直接影响了应对问题的不同方案。

比如我举个例子，楼房单元大门的门锁坏了，一直叫，这是一个具体的客观的问题。但是对这个问题的认识不同，可能就应对着不同的解决方案。

如果你认为这个问题是噪音问题，那么把报警的蜂鸣器切断就解决问题了。如果你认为问题就是锁坏了，那么就去找人修好锁，就解决问题了。如果你认为问题是有人蓄意破坏，那么报警找到破坏锁的人，让他赔偿，修好门锁，并保证不再犯，就是解决问题了。

如果你认为锁经常坏，是因为住户觉得开单元门的锁很麻烦，并非纯粹恶意破坏，那么就去给单元门升级智能锁，让它支持面部识别开锁、NFC 门卡开锁、密码开锁、手机 App 遥控一次性开锁等方便实用的功能，就非常好地解决问题了。

同时，你也可以在单元门口增加监控，提示“锁具昂贵，单元门在监控范围内，破坏锁具涉嫌犯罪！”。并附带一个物业电话，提示物业可以帮快递人员等临时进门的人员开门，帮业主新增解锁的面部 ID、办理 NFC 门卡等事情，那就更加完美地解决问题了。

架构师需要从实际问题出发，给出自己对问题的根本理解。所以，架构师和程序员相比，是一个更具有个性化的职业，也是一个更考验自己经验和技术功底的职业。而架构师最重要的能力之一呢，就是对问题的理解的深度。理解问题的这种能力，除了每个人的天赋之外，更多的是依靠你反复沟通、反复思考，以及在某个行业和领域的经验积累。

随着自己的技术积累、反复的思考演练以及过往的实际经验，架构师会慢慢形成自己的架构理念。这个理念，就会进一步形成架构师自己认识问题和理解问题的风格。当然，这些

都是发生在架构师心里的事情，在外人看来，这时候架构师还没有任何实际的产出。那么，架构师的工作成果是什么呢？

架构师的输出

架构师工作的输出，其实有两部分。一部分是自己对问题的理解和定义，另一部分就是给出解决问题的框架模型，也就是软件架构。

问题的定义

对问题的定义，不是简单的文字性的描述。换句话说，定义可以是从系统架构角度给出的需求分析，或者说是对系统的建模。说起建模，你可能会觉得比较高深，不好理解。我们也可以称之为高层设计（high-level design）。

很多中间件和产品的建模会抽象一些，里面的模块也会有着架构师自己深深的特色烙印。比如 Kafka，将系统抽象为 borker、partition、message、offset、consumer group 等关键模块，通过这些模块的互动，组合成一个独具特色的消息系统。Kafka 里面的各个模块，已经非常抽象，和具体的业务没有了直接的关系。

在 Kafka 的架构师看来，消息队列和 log 系统是相通的。正是这种独到的、富有洞察力的见解，才造就了 Kafka 这种独特的架构。也正是这种对问题本质的理解，让 Kafka 很好地解决了大数据浪潮下传统消息队列无法应对的问题，让 Kafka 在消息队列市场上独树一帜，赢得了很大的市场份额。

所以说，对问题的定义很重要。能够给出直指问题要害的定义，是非常考验一个架构师，尤其是偏产品类架构师的功力的。对于很多产品来说，架构师能够给出一个准确的定义，就已经成功了一小半，哪怕这时候一行代码都没有。

解决问题的模型

如果说给出问题的定义，更多的是依靠架构师对行业理解，那么给出解决问题的模型，就更多的是考验架构师的技术实力了。这时候架构师要给出的，是每个模块的详细设计。包括每个模块的接口定义、技术选型、关键实现的设计等等。

对于 Kafka 来说，如何处理这个消息队列系统的各种技术问题，就是架构师接下来要解决的问题。Kafka 的理念虽然非常简约，但是作为一个可靠的消息中间件，Kafka 本身是一

个非常复杂的系统。

这就好像 E=MC2 一样。质能转换方程很简单，但是造出一个原子弹来，却是一个非常复杂的工程，也和理论研究是两个维度的东西。这里我们不去展开说 Kafka 的架构，从架构师的责任上来说，构建的这个模型，要能够落地到具体的模块、接口、功能。这就是系统架构师要给出的详细设计。

对于比较庞大的系统，做高层设计的架构师，可能并不是每个模块的架构师。所以架构师们也不是都在做同一层面上的事情。

如果要形容一个架构师的核心能力，那会是两个词：积累和眼界。积累来自于架构师实操过的各种产品、搜集到问题后的思考、对行业各种技术的掌控、结合实际情况在心中进行的反复演练等等。眼界是一个架构师对行业和技术的产生的自己的理解。这两者决定了一个架构师的能量。

如何一步步成为架构师

那么我们软件工程师，如何才能一步步地成长为软件系统架构师呢？其实在我看来，任何一个软件工程师，从写第一行代码的时候开始，就已经既是软件工程师，又是软件系统架构师了。因为我们在写代码之前，肯定要经历这么一个理解问题、分析和定义问题、构建系统的过程。知道自己要干什么，要怎么干，最后一步才是写代码。

架构师的成长之路，就是在这个路径上一步步有侧重点地走。但侧重点则正好是反过来的。

夯实技术实力

首先你得重视自己写代码的能力，打好学习技术的基础。在这个阶段，程序员要能写得一手好代码，把基础的知识都学牢固，比如数据结构、网络、操作系统等等，建立起自己的技术领地。在写代码中，慢慢开始理解程序设计的技巧，比如设计模式、模块化、高内聚低耦合等等。

别看前面说架构师的输入和输出部分有些虚，感觉不会技术的人也能搞。但实际情况是，一个合格的架构师，肯定是要有深厚的技术做支撑的。之所以架构师可以搞这些看上去“虚头巴脑”的东西，还能让这些东西落地，就是因为技术够硬，心里有底。

随着我们程序员经验的积累，我们会慢慢发现，写代码是整个过程中最容易的一步。这也是为什么我会在外包和外派相关的章节中说，做普通的外包和外派，如果只是给甲方写代码，那么成长是很快会看到天花板的。因为工作性质没有给这些程序员向下一个阶段发挥的机会。

注重架构师能力培养

紧接着，就要更进一步，开始注重自己看问题、理解问题、分析问题的能力。简单来说，就是把重心向前挪，理解代码背后解决的问题，理解代码背后的设计理念，主动思考这些“虚头巴脑”的问题。

这个阶段，可以从熟悉自己做的系统的架构设计开始。也可以找一些优秀的开源框架，学习它们的架构设计。比如 Java 中最常见的 slf4j，就是一套设计得非常优秀的日志系统。几乎每个 Java 程序员都在用，但是，我们去深入理解过它的架构吗？在 slf4j 的世界里，它是如何定义“记日志”这个事情的？它又是使用了哪些技巧，让这个日志框架可以灵活高效呢？

同时，这个阶段程序员也要开始主动承担起，一些小模块的设计工作。系统架构师是一个需要非常多的实践积累的工作。每个人都有自己的成长方式，但是多做设计多思考，是普通人进阶软件系统架构师的不二法门。

保持开放的心态

软件系统架构师要有一个开放的心态，不能固守己见。优秀的系统架构，需要跟上发展，打破常规。比如前面提到的 Kafka 的设计，我当时第一次看到的时候，不由得心里说：我靠，这玩意有意思，对味儿。

再举个简单的例子。Chrome 在飞速抢占市场份额的那几年，曾经有一个大的变化，那就是让每个页面都是一个进程。要知道，之前多标签的浏览器，都是共享一个进程的。我当时得知这个变化，看着我的破电脑，心里想，Chrome 这是脑子进水了么？

现在回过头来想，我这种用着破电脑的小程序员，和 Chrome 的架构师的境界果然就是不一样啊。每个页面一个进程，一方面解放了 JS engine，让不同的标签之间不再受制于同一个 JS engine。另一方面这种变化也代表了浏览器发展的未来方向：因为一个标签，它

就是一个单独的应用程序啊。这是 Chrome 对标签的崭新定义，只是我等俗人要过很多年才能体会到的。

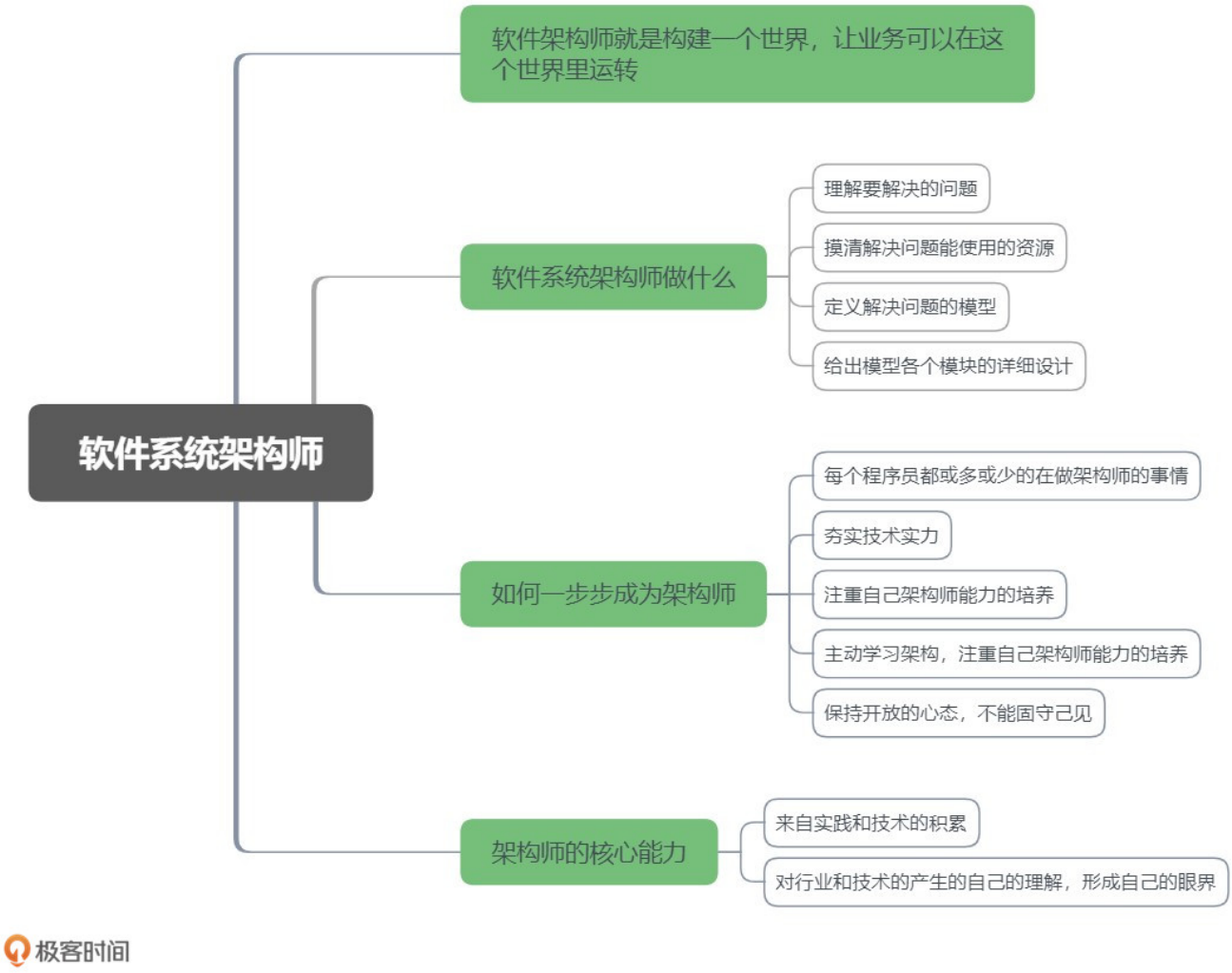
我们在实践系统架构的时候，也不要怕推翻自己之前的设计。你应该这么想：如果对于一件事情，我今天的认识和昨天是一样的，那就代表我这一天没有成长，如果我今天的认识和去年是一样的，那就代表我这一年没有成长。你想想，是不是这个道理？

总结

软件架构师，是要从无到有去设计出一个“新的世界”，制定这个世界的各种规则，让业务在自己设计的这个世界里运转起来。而这一切都需要实际经验的积累，这种积累需要的条件，远比单纯的技术积累复杂，所以架构师是一个需要岁月沉淀的工作。

对于程序员来说，不是年纪大了没人要，而是能力没跟上年纪的增长，才会没人要。程序员做架构，不需要有架构师这么一个名头。因为如果我们自己做一些业余项目，肯定就是先开始思考架构设计。如果是工作中安排的编码工作，其实只要自己平时写代码的时候多思考，多想想代码背后的业务，代码背后的系统设计，慢慢也会积累一些系统架构相关的经验。

当然，更重要的是去实践。工作中没机会，可以自己搞一些业余项目实践。自己想的可能很好，真的实际去做做看，可能会发现自己想的东西有很多纰漏。而系统架构的经验，就是这样一点点积累起来的。



思考题

关于软件架构师这个职位你有哪些疑问呢？你在工作中有承担过软件架构师的职责吗？你遇到过哪些优秀的软件架构师呢？

好，今天的课程就结束了，希望可以帮助到你，也希望你在下方的留言区和我参与讨论，并把文章分享给你的朋友或者同事，一起交流一下。

提建议

更多课程推荐

© 版权归极客邦科技所有，未经许可不得传播售卖。 页面已增加防盗追踪，如有侵权极客邦将依法追究其法律责任。

上一篇 24 | 技术观：程序员在技术的成长之路上，有哪些陷阱？

下一篇 26 | 系统集成：为什么最容易出问题的是系统集成？

精选留言 (4)

写留言

Sdylan
2020-07-13

如何脱离问题而构建出来的架构，不要去看，基本都是费的。好的架构都是从业务问题来的，然后再抽象出来。一个很好的例子就是Kafka。看完此文章后，也开始慢慢理解公司设计出来的内部架构，以前觉得这玩意不就是那Spring系列，值得这么整么。其实发现是自己对业务的理解过于狭窄，随着接触的业务多了，开始慢慢明白为啥这么做。总之，架构师首先是业务架构师，本质上是业务与架构的tradeoff，关键是对问题思考和抽象层...
展开

作者回复:
有内味了有内味了～！

所以你也理解为什么软件工程师有饭吃吧，个人的理解，不是为了重复造轮子，虽然都是轮子，

但是对于每辆飞速行驶的车，对轮子上各种细节的雕琢，都可能会对业务带来巨大的提升，这辆业务的车是在沙漠里开，还是在湿路上开，还是在山路上开，都会对轮子提出非常具体的要求。甚至沙子是细沙还是粗沙，都不一样。所以只要全世界的业务还都没挪到软件系统上，只要业务还在发展，软件工程师就一直有饭吃。

当然，重新造轮子，得到最大锻炼的就是架构师。架构师要珍惜珍惜再珍惜每一个重新造轮子的过程，深入理解公司业务的需求，构建业务模型，打造能帮助公司飞驰的轮子。这种公司花钱出人让人得到实践锻炼的机会，遇到了做梦都能笑醒。

5

有学识的兔子
2020-07-14

1. 我有个疑问，有哪些方式可以让自己更好地获得业务信息；有哪手段可以帮助自己熟悉业务；熟悉业务之后怎么指导自己工作并提升工作业绩呢？

2. 我不是构架师，但觉得自己承担了一些相关的职责；例如：基于理解架构的设计前提...
展开

作者回复: 你貌似没写完哈哈哈。

获取业务的信息，先从自己做的系统入手。虽然我们平时最多的工作可能就是写代码。但是我们要想理解业务，还是要能跳出代码的世界，理解自己所做的系统解决了业务的什么问题。这个可能要具体系统具体分析，但是很多时候，我们所做的系统都远比我们自己想象的要更重要。

很多时候，正是像你所说的，“骨干架构简洁，即使后续各种原因堆砌了“烂代码”，对系统的影响仍然是局部；还有对业务的适配比较好，设计能对应业务，基于此开发，难度不高；”。这时候，业务的精髓都在架构里了，而业务开发很多时候只要做一些外围的事情，无需理解全局也能产出成果。但是，这并不是对自己的发展有利的事情。

我还是回到做饭的例子。核心的架构和系统搭建好了，就好像有了一个自动炒菜的机器。平时做业务代码，就是洗菜切配，按照菜谱喂给自动炒菜机。然后菜就做好了。感觉好像做菜就是这么简单，但其实真正核心的业务并不是洗菜切配，而是这个自动炒菜机。

如果要熟悉业务，我们就要克服自己的惰性，顺藤摸瓜也好，打破砂锅问到底也好，把自己系统的业务吃透。我可以给你一个方法你试试看。就是顺着数据这条路走。摸清自己业务的数据从哪里来的，分别代表什么含义，经过了哪些系统处理，这些系统又对数据做了哪些事情。

然后就是自己的系统，摸清数据在自己系统里的各个流向和处理方式。数据最终落地到了哪里，或者返回给了哪些系统。

最后就是再顺着数据走，看看这些落地的数据或者返回给别的系统的数据，是如何被使用的。

2

王智
2020-07-14

从大学时期就有一个思想：写代码其实就是创建一个世界，这个世界怎么创建，完全是由你来决定的。直到现在我依然这么觉得，只不过毕业工作近两年，感觉自己的进步不是很大，接下来该静下心来思考思考了。

作者回复:
打好技术基础，这是构建世界的基本能力。

1

Eden Ma
2020-07-18

哈哈遇到过只虚头巴脑，但是一直无法落实，还把责任甩给组员说是他们太菜，老写一些烂代码，还说历史遗留问题太多无法解决，需要一个得力助手让公司招，还谁都瞧不上，半年也招不到的架构师。

作者回复: 架构师要做的是根据资源解决问题的事情，不是列出问题强调难度、推卸责任的事情。

历史遗留问题，人员水平问题（如果真的是的话），都是架构师需要考虑的问题。架构师最没资格说这有问题那有问题。

啥都没问题还要架构师干嘛。

2